ELSEVIER
Theoretical Computer Science 161 (1996) 123–140
Theoretical Computer Science

On resource-bounded instance complexity☆

Lance Fortnow[a,1], Martin Kummer[b,*]

[a] *Department of Computer Science, University of Chicago, 1100 E. 58th St., Chicago, IL 60637, USA*
[b] *Institut für Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe, D-76128 Karlsruhe, Germany*

Received August 1994; revised March 1995
Communicated by O. Watanabe

Abstract

The instance complexity of a string x with respect to a set A and time bound t, $ic^t(x : A)$, is the length of the shortest program for A that runs in time t, decides x correctly, and makes no mistakes on other strings (where "do not know" answers are permitted). The instance complexity conjecture of Ko, Orponen, Schöning, and Watanabe (1986) states that for every recursive set A not in P and every polynomial t there is a polynomial t' and a constant c such that for infinitely many x, $ic^t(x : A) \geqslant C^{t'}(x) - c$, where $C^{t'}(x)$ is the t'-time bounded Kolmogorov complexity of x. In this paper the conjecture is proved for all recursive tally sets and for all recursive sets which are NP-hard under honest reductions, in particular it holds for all natural NP-hard problems. The method of proof also yields the polynomial-space bounded and the exponential-time bounded versions of the conjecture in full generality. On the other hand, the conjecture itself turns out to be oracle dependent: In any relativized world where P = NP the conjecture holds, but there are also relativized worlds where it fails, even if C-complexity is replaced by Sipser's CD-complexity. Additionally it is proved that the instance complexity measure is noncomputable and it is investigated whether for every polynomial t there is a polynomial t' such that $C^{t'}$-complexity is bounded above by CD^t-complexity.

1. Introduction

Instance complexity was introduced by Ko et al. [11, 18] (see also [13, Section 7.3.3]) as a measure of the complexity of individual instances of a decision problem

☆ An extended abstract of this paper appeared in the *Proc. STACS 95*, Lecture Notes in Computer Science, Vol. 900 (Springer, Berlin, 1995) 597–608.

* Corresponding author. E-mail: kummer@ira.uka.de.

[1] Partially supported by NSF Grant CCR 92-53582.

SSDI 0304-3975(95)00097-6

A. Intuitively, the *instance complexity* $ic(x : A)$ of x with respect to A is the length of the shortest program p which correctly computes $\chi_A(x)$ and does not make any mistakes on other inputs (it is permitted to output "do not know" answers). In this paper we consider the resource-bounded version $ic^t(x : A)$ where the running time of p is bounded by some polynomial t.

The trivial program which contains an encoding of x shows that $ic(x : A)$ is bounded by the Kolmogorov complexity of x. This fact can be transferred to the time-bounded setting. An instance x is called hard if this trivial upper bound is already optimal, i.e., there is no easier way to decide x than to explicitly encode x into the program. The "instance complexity conjecture" of Orponen et al. [18] states informally that every complex set has infinitely many hard instances. More precisely, in the polynomial-time bounded setting it is conjectured that every recursive set A not in P must have *p-hard instances*, i.e., for every polynomial t there is a polynomial t' such that $ic^t(x:A) \geqslant C^{t'}(x) + O(1)$ for infinitely many x, where $C^{t'}(x)$ denotes the t'-time bounded Kolmogorov complexity of x.

In this paper we prove several natural special cases of the conjecture and provide relativized counterexamples which show that our results are essentially optimal with regard to relativizing proof techniques.

We first show that the conjecture holds for all recursive tally sets which illustrates our basic proof technique. As a corollary we obtain the previous results of [18] as well as a proof of the conjecture for p-bi-immune sets, sets with co-sparse complexity cores, and leftcut sets. Of particular interest is the class of NP-hard sets. In this direction Orponen et al. [18] proved that every set which is NP-complete under honest m-reductions has p-hard instances, unless $E = NE$. We obtain a strong improvement of this result: All sets which are NP-complete under honest Turing reductions have p-hard instances, unless $P = NP$. In [18] a weak form of the conjecture is shown where the time bounds depend on the complexity of A. We show that the dependence on A can be removed. It also follows that the polynomial-space bounded and the exponential-time bounded version of the conjecture hold in full generality.

The instance complexity conjecture cannot be settled with relativizable methods, since we construct relativized worlds where it holds and where it fails. In fact, it may even fail for sparse sets, p-immune sets, and p-selective sets which shows that there is not much room for improving our results above by relativizing techniques. We also show that the *CD*-version of the conjecture – where C-complexity is replaced by Sipser's *CD*-complexity – fails in some relativized world. This is interesting because *CD*-complexity is much closer to instance complexity than C-complexity.

In addition, we show that the *ic*-measure is not computable confirming another conjecture from [18].

Finally, we compare time-bounded C- and *CD*-complexity and investigate whether for every polynomial t there is a polynomial t' such that $C^{t'}$-complexity is bounded above by CD^t-complexity.

2. Notation and definitions

For general background and unexplained notions we refer the reader to the textbooks [2, 3, 13, 19]. For the convenience of the reader we recall a few definitions that will be used later.

We consider strings over the alphabet $\Sigma = \{0, 1\}$. The length of a string x is denoted by $|x|$; ε is the empty string. The characteristic function of A is denoted by χ_A. If f is a function we write $f(x)\downarrow$ to denote that f is defined for argument x, and $f(x)\uparrow$ if $f(x)$ is undefined. PF is the class of all functions $f : \Sigma^* \to \Sigma^*$ that can be computed in polynomial time.

A function f is *honest* if there is a polynomial r such that $(\forall x, y)[f(x) = y \Rightarrow |x| \leqslant r(|y|)]$. f is an honest p-time m-reduction if $f \in$ PF and f is honest. A is reducible to B by an honest p-time Turing reductions if there are a polynomial r and a polynomial-time bounded oracle Turing machine M such that $\chi_A = M^B$ and for all x, y, if y is queried on input x, then $|x| \leqslant r(|y|)$, i.e., the length of the input has to be within a fixed polynomial of the length of any query.

A set A is *tally* if $A \subseteq 0^*$; A is *sparse* if there is a polynomial r such that $|A \cap \Sigma^n| \leqslant r(n)$; A is *co-sparse* if $\overline{A}$ is sparse. Let SPARSE denote the class of all sparse sets.

If A is recursive, then C is called a *complexity core* for A if C is infinite and for every deterministic machine M that accepts A and every polynomial r there are at most finitely many $x \in C$ such that the number of steps of M on x is bounded by $r(|x|)$. A is *p-immune* if it is a core for itself, and *p-bi-immune* if both A and $\overline{A}$ are cores for A.

A set A is called a *leftcut set* if there is an infinite string $r \in \Sigma^\omega$ such that $A = \{x \in \Sigma^* : x < r\}$. Here $<$ is the dictionary ordering of strings with 0 less than 1.

A set A is *self-reducible* if there is a polynomial-time bounded oracle Turing machine M such that $M^A(x) = \chi_A(x)$ and M^A with input x queries only strings of length less than $|x|$. A is *d-self-reducible* if $M^A(x)$ accepts iff at least one of the queries is answered positively.

$\mathrm{E} = \bigcup_{c>0} \mathrm{DTIME}(2^{cn})$ and $\mathrm{NE} = \bigcup_{c>0} \mathrm{NTIME}(2^{cn})$. UP is the class of all languages which are accepted by nondeterministic polynomial-time Turing machines with unique accepting computations. FewP is the class of all languages which are accepted by nondeterministic polynomial-time Turing machines M for which there is a polynomial r_M such that for all inputs x there are fewer than $r_M(|x|)$ accepting computations of M on x.

Our machine model is the class of deterministic Turing machines with three input tapes and an arbitrary number of work tapes. They compute partial functions from $\Sigma^* \times \Sigma^* \times \Sigma^*$ to $\Sigma^* \cup \{\perp\}$. If M is such a machine we denote the output of M on input (p_1, p_2, x) by $M(p_1, p_2, x)$, and the number of steps in the computation by $time_M(p_1, p_2, x)$. We assume that $t(n) > n$ for all time bounds $t = t(n)$.

The following notion of time bounded Kolmogorov complexity was introduced by Hartmanis [6], Ko [9], and Sipser [21]. Intuitively, the t-bounded Kolmogorov complexity of x is the length of the shortest program which computes x in $t(|x|)$ steps from the empty input.

Definition 1. For any time bound t and $x, y \in \Sigma^*$ the *t-bounded Kolmogorov complexity* of x conditional to y using M is defined as

$$C_M^t(x|y) = \min\{|p| : M(p, \varepsilon, y) = x \wedge time_M(p, \varepsilon, y) \leqslant t(|x|)\}.$$

The t-bounded Kolmogorov complexity of x using M is defined as

$$C_M^t(x) = C_M^t(x|\varepsilon).$$

Instance complexity was introduced by Ko et al. [11]. Intuitively, the t-bounded instance complexity of x with respect to A is the length of the shortest program which runs in time $t(n)$, correctly decides whether x is in A, and does not make mistakes on any other input (where it is allowed to output $\perp$ for "do not know").

Definition 2. Let $A \subseteq \Sigma^*$. A function $f : \Sigma^* \to \{0, 1, \perp\}$ is called *A-consistent* if for all $x \in dom(f), [f(x) = \chi_A(x) \vee f(x) = \perp]$.

For any time bound t the *t-bounded instance complexity* of $x \in \Sigma^*$ with respect to A using M is defined as

$$ic_M^t(x : A) = \min\{|p| : \lambda z . M(p, \varepsilon, z) \text{ is } A\text{-consistent}, (\forall z)[time_M(p, \varepsilon, z) \leqslant t(|z|)], \text{ and } M(p, \varepsilon, x) = \chi_A(x)\}.$$

The *CD*-version of Kolmogorov complexity was introduced by Sipser [21]. Intuitively, the *CD*-complexity of x is the shortest program which accepts x and rejects every other string. This notion is only of interest in the time-bounded setting, since its unbounded version coincides with unbounded Kolmogorov complexity.

Definition 3. For any time bound t and $x, y \in \Sigma^*$ the *t-bounded CD-complexity* of x conditional to y using M is defined as

$$CD_M^t(x|y) = \min\{|p| : \lambda z . M(p, y, z) = \chi_{\{x\}} \text{ and } (\forall z)[time_M(p, y, z) \leqslant t(|z|)]\}.$$

The t-bounded *CD*-complexity of x using M is defined as

$$CD_M^t(x) = CD_M^t(x|\varepsilon).$$

Thus, unconditional *CD*-complexity can be considered as a special case of instance complexity; for all natural interpreters M we have $CD_M^t(x) = ic_M^t(x : \{x\})$.

There is a universal Turing machine U (an "optimal interpreter") such that the following invariance property holds (see [6, 18, Theorem 2.1]).

Fact 4. *For every Turing machine M there is a constant c such that for all sets A, all time bounds t, and all strings x, y :*

$$ic_U^{t'}(x : A) \leqslant ic_M^t(x : A) + c,$$

$$C_U^{t'}(x|y) \leqslant C_M^t(x|y) + c,$$

$$CD_U^{t'}(x|y) \leqslant CD_M^t(x|y) + c,$$

where $t'(n) = ct(n) \log t(n) + c$.

In the following we fix such a U and write C^t, ic^t, CD^t for C_U^t, ic_U^t, CD_U^t. If the context is clear we may write $U(p)$ instead of $U(p, \varepsilon, \varepsilon)$, and $U(p, y)$ instead of $U(p, \varepsilon, y)$, etc.

3. The instance complexity conjecture

The following basic fact shows that the Kolmogorov complexity is an upper bound for the instance complexity.

Fact 5. (Orponen et al. [18, Proposition 3.1]). *For any time constructible function t there is a constant c such that for any set A and string x,*

$$ic^{t'}(x : A) \leqslant C^t(x) + c,$$

where $t'(n) = ct(n) \log t(n) + c$.

Clearly, if $A \in \mathrm{DTIME}(t)$ then ic^t is bounded by a constant. Orponen et al. [18] conjectured that for $A \notin \mathrm{DTIME}(t)$, the instance complexity must be infinitely often as high as the Kolmogorov complexity.

Conjecture 6 (Orponen et al. [18]). Let t be time-constructible and A recursive. If $A \notin \mathrm{DTIME}(t)$ then there is a constant c and infinitely many x such that

$$ic^t(x : A) \geqslant C^{t'}(x) - c,$$

where $t' = \mathrm{O}(t \log t)$.

The following observation characterizes the class P.

Fact 7. (Orponen et al. [18, Proposition 3.2]). *A set A is in* P *iff there is a polynomial t and a constant c such that for all x, $ic^t(x : A) \leqslant c$.*

Definition 8 (Ko [10], Orponen et al. [18]). A set A has *p-hard instances* if for every polynomial t there exists a polynomial t' and a constant c such that for infinitely many $x, ic^t(x : A) \geqslant C^{t'}(x) - c$.

This motivates the following interesting special case of Conjecture 6, which is to a larger extent independent of the machine model.

Conjecture 9. Every recursive set $A \notin \mathrm{P}$ has p-hard instances.

Clearly, Conjecture 9 follows from Conjecture 6. By definition, the CD-complexity is an upper bound for the instance complexity in the following sense.

Fact 10. *For any time bound t, any set A, and all strings x,*

$$ic^{t'}(x : A) \leqslant CD^{t}(x) + \mathrm{O}(1),$$

where $t' = \mathrm{O}(t \log t)$. (For natural optimal interpreters we even have $t' = t + \mathrm{O}(1)$.)

This motivates the following CD-version of Conjecture 9.

Conjecture 11. For every recursive set $A \notin \mathrm{P}$ and every polynomial t there is a polynomial t' and a constant c such that

$$ic^{t}(x : A) \geqslant CD^{t'}(x) - c \text{ for infinitely many } x.$$

The CD-complexity is less than or equal to the Kolmogorov complexity; for every polynomial t there is a polynomial t' such that for all x, $CD^{t'}(x) \leqslant C^{t}(x) + \mathrm{O}(1)$ (cf. [13, Theorem 7.2]). Thus, Conjecture 11 is weaker than Conjecture 9.

Remark. Orponen and his coworkers [17, 18] stated an unbounded version of the instance complexity conjecture for all nonrecursive r.e. sets. It is shown in [12] that this conjecture fails in general. However, it can be established for certain r.e. complete sets [4, 12].

3.1. Positive results

As partial evidence for Conjecture 9, Orponen et al. [18] show that every E-complete set has p-hard instances, and SAT has p-hard instances unless E = NE (see [18, Corollaries 5.6 and 5.7]). Both results can be deduced from the following new[1] theorem. In the proof we introduce the basic construction on which we build later.

Theorem 12. *Every recursive tally set $A \notin \mathrm{P}$ has p-hard instances.*

Proof. Assume that $A \subseteq 0^*$ is a recursive tally set and $A \notin \mathrm{P}$. We first present an informal outline of the construction. Since A is tally we know that the potentially hard instances are in 0^*. The idea is to build programs p which, on empty input,

[1] The theorem does not follow from [18, Theorem 5.5] since their result does not apply to p-immune sets, but there exist recursive p-immune tally sets.

compute strings 0^n in $t'(n) = O(n^3 t(n))$ steps and diagonalize against all programs q with $|q| \leqslant |p|$ that might witness $ic^t(0^n : A) \leqslant |p|$. Since $C^{t'}(0^n) \leqslant |p|$, this implies that $ic^t(0^n : A) \geqslant C^{t'}(0^n)$, as required. For the diagonalization we want to ensure that $U^t(q, 0^n) = \perp$ or $U^t(q, 0^n)\uparrow$, for all A-consistent $q, |q| \leqslant |p|$. To this end we run a slow simulation of the decision procedure of A, so that eventually all A-inconsistent programs can be eliminated. Simultaneously we check for larger and larger n whether $U^t(q, 0^n) = \perp$ or $U^t(q, 0^n)\uparrow$ for all $q, |q| \leqslant |p|$ that currently appear to be A-consistent. From some stage on we will be considering only A-consistent programs q, though during the construction we never know for sure when this happens. If a suitable n is not found, then for almost all n there is an A-consistent $q, |q| \leqslant |p|$ such that $U^t(q, 0^n) = \chi_A(0^n)$. In this case we can amalgamate the finitely many q's and obtain a new program that decides A in polynomial time, contradicting the hypothesis. Thus, a suitable n is eventually found, and the program p terminates. Note that though 0^n is a hard instance its Kolmogorov complexity may be very low.

Now we turn to the formal details. Let t be a given polynomial of the form $n^k + k$, and let N be a Turing machine which computes χ_A. For $q, \sigma \in \Sigma^*$, we say that q is *t-compatible with σ* if for all $m \leqslant |\sigma|$,

$$U^{t(|x_m|)}(q, x_m)\downarrow \ \wedge\ [U^{t(|x_m|)}(q, x_m) \neq \perp \implies U^{t(|x_m|)}(q, x_m) = \sigma(m)].$$

Here x_m is the mth string in the standard ordering and $\sigma(m)$ denotes the mth bit of σ. Note that $\lambda x.U^{t(|x|)}(q, x)$ is total and A-consistent iff q is t-compatible with every finite initial segment σ of χ_A. We define an algorithm M to witness that $C^{t'}_M(x) \leqslant ic^t(x : A)$ for some polynomial t' and infinitely many $x \in 0^*$.

M computes as follows for input p:
Let $I = \{q \in \Sigma^* : |q| \leqslant 2 \cdot |p|\}$. Let $n = |I|$. Goto stage $n + 1$.
Stage n:

(1) Spend n steps in computing $N(x_i)$ for $i = 1, 2, \ldots$, and let σ be the maximal initial segment of χ_A which has been computed in this way.
(2) Eliminate all q from I which are not t-compatible with σ.
(3) Compute $U^{t(n)}(q, 0^n)$ for all $q \in I$. If one of these values is in $\{0, 1\}$ then goto stage $n + 1$. Else let $M(p) = 0^n$ and halt.

The correctness is verified as follows:

(a) $M(p)$ terminates for all p: Suppose for a contradiction that $M(p)$ does not terminate. Then n increases infinitely often and σ denotes larger and larger initial segments of χ_A. Let I_0 denote the final value of I, and let n_0 be the stage when this final value is reached. Since σ is unbounded it follows that for all $q \in I_0$ and all m, if $U^{t(m)}(q, 0^m) \in \{0, 1\}$ then $U^{t(m)}(q, 0^m) = \chi_A(0^m)$. Since no stage $n \geqslant n_0$ terminates it follows that for every $n \geqslant n_0$ there is $q \in I_0$ with $U^{t(n)}(q, 0^n) = \chi_A(0^n)$. Thus, if we amalgamate the programs in I_0 and patch the finitely many arguments 0^m, $m < n_0$, we obtain an $O(t)$-time bounded algorithm for A, so $A \in P$, a contradiction.

(b) If $M(p) = 0^n$ then the computation of M on input p takes only $O(n^3 t(n))$ steps. This is obvious since the computation in each stage $m \leqslant n$ uses $O(m+m|I|t(m)+|I|t(m))$ steps and $|I| \leqslant n$.

(c) There are infinitely many n such that $M(p) = 0^n$ for some p.

(d) If $M(p) = 0^n$ then $ic^t(0^n : A) > 2 \cdot |p|$. This is clear, since the algorithm terminates only if $U^{t(n)}(q, 0^n) \notin \{0, 1\}$ for all A-consistent q with $|q| \leqslant 2 \cdot |p|$.

Hence there is a polynomial t' such that for infinitely many n : $ic^t(0^n : A) > 2 \cdot |p| \geqslant 2 \cdot C_M^{t'}(0^n)$. By invariance, there is a polynomial t'' and a constant c such that for all x, $C^{t''}(x) \leqslant C_M^{t'}(x) + c$. Thus, $ic^t(0^n : A) > C^{t''}(0^n)$ for infinitely many n. □

Remark. Note that with a slight modification we can even show that for every recursive function f there are infinitely many n such that $ic^t(0^n : A) > f(C^{t''}(0^n))$. To achieve this $M(p)$ first computes $f(|p|)$ and initializes $I = \{q : |q| \leqslant 2|p| + f(|p|)\}$ and $n = |I| +$ (number of steps to compute $f(|p|)$). The rest of the construction is identical. This result stands in contrast to Fact 5 which states that the Kolmogorov complexity is an upper bound of the instance complexity. However, time-bounded Kolmogorov complexity is very sensitive to small changes of the time bound and if we slightly increase the time bound, as it happens above where $t'' > t$, then the Kolmogorov complexity may decrease such that the old values are no longer within any recursive function of the new ones. In comparison with the absolute nature of unbounded Kolmogorov complexity this is somewhat pathological.

We now present several corollaries of Theorem 12 and its proof. The following Lemma is proved in [18, Lemma 5.8].

Lemma 13. *If A has p-hard instances and $A \leqslant_m^p B$ by an honest reduction, then B has p-hard instances.*

Remark. Orponen et al. [18] claimed a stronger form of Lemma 13 without the honesty condition, but their proof requires honesty, and in Corollary 20(a) we construct a relativized world where honesty is necessary.

The proof in [18] uses the hypothesis that for all $f \in \mathrm{PF}$ there is a constant e such that for any polynomial t there is a polynomial t' such that for all $x, C^{t'}(f(x)) \leqslant C^t(x) + e$. While this is certainly correct for any *honest* $f \in \mathrm{PF}$, it can be shown that it does not hold in general, even if f is 1–1. The reason is that the time bound t' measured on $|f(x)|$ can be much smaller than t on $|x|$, if f is not honest. The statements of Lemma 5.8 and Corollaries 5.9 and 5.10 in [18], should therefore be restricted to honest reductions.

Corollary 14 (Orponen et al. [18]). *Every* E*-complete set has p-hard instances.* SAT *has p-hard instances unless* E = NE.

Proof. It is well-known that there exists a tally set $A \in \mathrm{E} - \mathrm{P}$. By a result of Berman and Watanabe (see [22]), A is m-reducible to any E-complete set by length increasing reductions. Since, by Theorem 12, A has p-hard instances, it follows by Lemma 13 that B has p-hard instances, too.

By a result of Hartmanis [7], if $\mathrm{E} \neq \mathrm{NE}$ then there is a tally set in $\mathrm{NP} - \mathrm{P}$ which in turn is reducible to SAT by a length increasing reduction. Thus, as above, SAT has p-hard instances. □

Remark. By a different approach, Buhrman and Orponen [4, Theorem 4.3] proved that every E-complete set has an exponentially dense subset of p-hard instances.

Corollary 15. *Let A be a recursive set. Each of the following properties implies that A has p-hard instances.*

(a) *A is p-bi-immune.*

(b) *A has a co-sparse complexity core.*

(c) *A is a leftcut set and $A \notin \mathrm{P}$.*

Proof. Assume that A is recursive.

(a) If A is p-bi-immune then $B = A \cap 0^*$ is a tally set not in P. By Theorem 12, B has p-hard instances and it easily follows that A has p-hard instances, too.

(b) Suppose that C is a co-sparse complexity core for A and let r be a polynomial such that $|C \cap \Sigma^n| \geqslant 2^n - r(n)$. We modify the construction of Theorem 12 and search the hard instances within the lexicographically first $r(n)+1$ strings of each length n, let this set be denoted by Γ_n. In step (3) we compute $U^{t(n)}(q,x)$ for all $q \in I$ and all $x \in \Gamma_n$. If an $x \in \Gamma_n$ is found such that none of the values $U^{t(n)}(q,x)$ is in $\{0,1\}$, then let $M(p) = x$ and halt. Otherwise, goto stage $n+1$. Clearly, the computation in stage n requires only $\mathrm{O}(n^2 \cdot r(n) \cdot t(n))$ many steps.

It suffices to show that $M(p)$ terminates, the rest is analogous as above. If $M(p)$ does not terminate, then the amalgamation of the A-consistent programs in I computes A correctly in polynomial time for almost all x in $B = \bigcup_n \Gamma_n$. Since $B \cap C$ is infinite and $B \in \mathrm{P}$, C is not a complexity core, a contradiction. Thus $M(p)$ terminates.

(c) Suppose that $A \notin \mathrm{P}$ is a leftcut set, as witnessed by the infinite string $r \in \Sigma^\omega$. There are recursive leftcut sets which are not even $\leqslant_{\mathrm{tt}}^p$-equivalent to any tally set [16]. Thus, the result does not follow from Theorem 12 and Lemma 13.

Instead we modify the construction of Theorem 12 as follows: In step (3) we try to find the lexicographically greatest $x \in \Sigma^n$ such that $x \in L$ (i.e., x is a prefix of r). If we had an oracle for A, this could be done with n queries using binary search. Instead of A we use the amalgamation of the programs in I for that purpose. If we query $z \in \Sigma^n$ and find that $U^{t(n)}(q,z) \notin \{0,1\}$ for all $q \in I$, then we let $M(p) = z$ and halt. Otherwise, we assume that the answer is $\min\{U^{t(n)}(q,z) : q \in I\}$ and continue the search. If the search terminates, say with $x = y_n$, then we go to stage $n+1$.

Again, it only remains to show that $M(p)$ is eventually defined. Otherwise, for almost all n, the answers given by the amalgamation are correct and y_n is indeed the

prefix of r of length n. Since y_n is computed in polynomial time in n, it follows that $A \in \mathrm{P}$, a contradiction. Thus, $M(p)$ terminates. □

Remark. Since p-bi-immune sets are not *meager in* E and have *measure* 1 *in* E (see [15] for the definitions and proofs), the same holds for sets with p-hard instances.

For NP-hard sets we can further exploit the idea from the proof of Corollary 15(c), to get the following strong improvement of Corollary 14.

Theorem 16. *Every recursive set A which is* NP*-hard w.r.t. honest p-time Turing reductions has p-hard instances unless $A \in \mathrm{P}$.*

Proof. For every polynomial t we let L^t be the set of all triples $(x, 0^n, I)$, $x \in \Sigma^*$, $n \in \mathcal{N}$, and I a finite subset of Σ^*, such that there is a string $z \in \Sigma^n$ which extends x and $U^{t(n)}(q,z) \notin \{0,1\}$ for all $q \in I$. Clearly $L^t \in \mathrm{NP}$. Suppose we are given L^t as an oracle. Then on input $(0^n, I)$ we can check whether there is a string $z \in \Sigma^n$ with $U^{t(n)}(q,z) \notin \{0,1\}$ for all $q \in I$, and if this is the case, then such a z can be computed by "prefix searching" (see [2, p. 61]) with n queries to L^t. The number of steps is bounded by a polynomial in $n + size(I)$.

Now assume that A is a recursive set which is NP-hard w.r.t. honest p-time Turing reductions and fix a polynomial t. Since $L^t \in \mathrm{NP}$, there is a polynomial-time bounded oracle Turing machine M_0 and a polynomial r such that $L^t = M_0^A$ and, for all x, y and all oracles X, if M_0^X queries y on input x, then $|x| \leqslant r(|y|)$.

We proceed as in the proof of Theorem 12, i.e., we define an algorithm M to witness that $C_M^{t'}(x) \leqslant ic^t(x : A)$ for some polynomial t' and infinitely many x. Let N be a decision procedure for A.

M computes as follows for input p:
Let $I = \{q \in \Sigma^* : |q| \leqslant 2 \cdot |p|\}$. Let $n = |I|$. Goto stage n.
Stage n:
(1) Spend n steps in computing $N(x_i)$ for $i = 1, 2, \ldots$, and let σ be the maximal initial segment of A which has been computed in this way.
(2) Eliminate all q from I which are not t-compatible with σ.
(3) Using L^t as an oracle, search for a string z of length n such that $U^{t(n)}(q,z) \notin \{0,1\}$ for all $q \in I$. Queries to L^t are answered by simulating $M_0(x, 0^n, I)$. If in this simulation a string y is queried, then answer according to $\min\{U^{t(n)}(q,y) : q \in I\}$. If this set does not contain 0 or 1, let $M(p) = y$ and halt. If the search terminates but z has not the required property, then goto stage $n + 1$. Else let $M(p) = z$ and halt.

$M(p)$ terminates for all p, unless $A \in \mathrm{P}$: Suppose that $M(p)$ does not terminate. After some stage n_0 all programs in $I = I_0$ are consistent with A. Consider any stage $n > n_0$. Let $H_n = \{z \in \Sigma^n : (\forall q \in I_0)[U^{t(n)}(q,z) \notin \{0,1\}]\}$. If $H_n \neq \emptyset$, then the prefix search finds an element $z \in H_n$. By the consistency of I, every answer to a query in the simulation exists (otherwise we terminate) and is correct. Thus, the only reason,

why stage n was not successful, is that $H_n = \emptyset$. Hence if we amalgamate all programs in I_0 and patch finitely many exceptions, we obtain $A \in \mathrm{DTIME}(t(n))$, i.e., $A \in \mathrm{P}$.

By the honesty of M_0, it follows that there is a polynomial t' such that if $M(p)$ terminates after s steps with output z, then $s \leqslant t'(|z|)$. Thus, $C_M^{t'}(z) \leqslant |p|$ and $ic^t(z : A) > 2|p|$.

So, if $A \notin \mathrm{P}$, then $M(p)$ terminates for all p and there are infinitely many z such that $C^{t''}(z) \leqslant ic^t(z : A)$ for some polynomial t''. □

Corollary 17. *Let $\mathscr{C} \supseteq \mathrm{NP}$ be any complexity class. Every set A which is $\mathscr{C}$-complete w.r.t. honest p-time Turing reductions has p-hard instances, unless $\mathscr{C} \subseteq \mathrm{P}$. In particular,* SAT *and* QBF *have p-hard instances unless* P = NP *or* P = PSPACE.

We can apply the construction from the proof of Theorem 12 to arbitrary recursive sets not in P; then we have to consider all x of a given length, hence the running time increases by an exponential factor. However, it does *not* depend on A. Thus we get the following improvement of [18, Theorem 5.1].

Corollary 18. *Let $t(n) \geqslant n$ be a nondecreasing time constructible function, and let A be a recursive set not in* DTIME(t). *Then there exists a constant c such that for infinitely many x, $ic^t(x : A) \geqslant C^{t'}(x) - c$, where $t'(n) = c2^{2n}t(n)(n + \log t(n))$.*

This shows that a corresponding version of Conjecture 9 holds for E and exponential time bounds: If A is a recursive set not in E then for every $t \in 2^{lin} = \{2^{cn} : c > 0\}$ there is $t' \in 2^{lin}$ such that for infinitely many x, $ic^t(x : A) \geqslant C^{t'}(x)$.

Similarly, the space bounded version of the conjecture holds, where P is replaced by PSPACE and ic^t, C^t by their space bounded analogs.

3.2. Relativized counterexamples

The next result shows that Conjecture 9 cannot be settled with relativizing techniques.

Theorem 19. (a) *If* P = NP *then Conjecture 9 holds (and this result relativizes).*

(b) *There is an oracle B such that Conjecture 9 (and hence also Conjecture 6) fails relative to B.*

Proof. (a) This follows at once from Theorem 16.

(b) This argument is based on Hartmanis' idea [6, p. 444] of constructing an oracle B with $\mathrm{P}^B \neq \mathrm{NP}^B$, using Kolmogorov complexity. The construction automatically yields $\mathrm{P}^B \neq \mathrm{UP}^B$ as was noticed in [14, Theorem 4.10].

Let $tow(0) = 1, tow(i) = 2^{tow(i-1)}$. Let x_i be a Kolmogorov-random string of length $tow(i)$ (i.e., $C(x_i) \geqslant |x_i|$), and let $B = \{x_i : i \geqslant 0\}$. Then for every polynomial t and almost all i we have $C^{t,B}(x_i) \geqslant |x_i|/2$: If x_i is computed by the universal oracle Turing machine U^B in $t(|x_i|)$ steps from a string p, then for sufficiently large i, the machine does not query any string of length $tow(j)$ for $j > i$. Thus we can compute x_i without

the oracle, if we are given $p, x_0, \ldots, x_{i-1}$ and the binary representation of the least number $s \leqslant t(|x_i|)$ such that $U^B(p)$ queries x_i in step s. This can be coded in a string of length $|p| + O(\log(|x_i|)) \geqslant C(x_i)$. Since x_i is random, it follows that $|p| \geqslant |x_i|/2$ for all sufficiently large i.

Now choose any set $A \subseteq B$ such that $A \in \mathrm{DTIME}^B(2^n) - \mathrm{P}^B$. Let $t(n) = n^2$, then there is a constant d such that $ic^{t,B}(x : A) \leqslant d$ for all $x \notin B$. Furthermore, $ic^{t,B}(x_i : A) \leqslant \log(|x_i|) + d$ for all i. This is witnessed by a machine which outputs $\perp$ for all $x \neq x_i$ and $\chi_A(x)$ for $x = x_i$. Note that a string of length $\log(|x_i|) + d$ suffices to specify $|x_i|$ and $\chi_A(x)$. Thus for every polynomial t', every constant c, and almost all x we have $ic^{t,B}(x : A) < C^{t',B}(x) - c$. □

Corollary 20. (a) *There is a relativized world in which p-hard instances are not inherited upwards under $\leqslant_1^p$-reductions.*

(*This is a relativized counterexample to* [18, Lemma 5.8].)

(b) *None of the following properties implies in all relativized worlds that a recursive set $A \notin \mathrm{P}$ has p-hard instances*:

(b1) *A is sparse.*

(b2) *A is p-immune.*

(b3) *A is p-selective.*

(b4) *A is d-self-reducible.*

(*Thus, we cannot improve Theorem 12 and Corollary 15 (a) and (c) in a relativizable way from "tally" to "sparse" or from "p-bi-immune" to "p-immune" or from "leftcut" to "p-selective", respectively. Also, using only d-self-reducibility does not suffice for showing that* SAT *has p-hard instances.*)

(c) *For every function $f(n)$ such that $f(n)/\log n$ is nondecreasing and unbounded there is a relativized world with $\mathrm{P} \neq \mathrm{NP}$ and for every $A \in \mathrm{NP}$ there is a polynomial t such that $ic^t(x : A) = f(|x|) + O(1)$ for all x.*

(*This shows that Theorem 4.1 of Orponen et al.* [18] *is optimal w.r.t. relativization and gives a relativized negative answer to a question of Ko* [10, *p.* 335] *who asked whether an* NP*-hard set must have an infinite number of hard instances which have high Kolmogorov complexity.*)

Proof. (a) Relative to the oracle B from the proof of Theorem 19(b), p-hard instances are not inherited upwards under nonhonest $\leqslant_1^p$-reductions:

Let $\tilde{H} \subseteq B$ such that $\tilde{H}$ is recursive in B and $\tilde{H} \notin \mathrm{E}^B$. By construction we get that $\tilde{H}$ and also $H = \{0x : x \in \tilde{H}\}$ do not have p-hard instances relative to B. Let $D = \{0^{nr(x)} : x \in H\}$ where $nr(x)$ is the number of x in the lexicographical ordering of all strings. Since $H \notin \mathrm{E}^B$, it follows that $D \notin \mathrm{P}^B$. As D is tally and recursive in B, Theorem 12 relativized to B implies that D has p-hard instances relative to B. Clearly, $D \leqslant_1^p H$ via $f \in \mathrm{PF}$ such that $f(0^{nr(x)}) = 0x$ and $f(z) = 1z$ for $z \notin 0^*$.

(b) Let A be the set constructed in the proof of Theorem 19(b). Note that A is sparse and p-selective (and also 1-cheatable [1]) relative to B. By a standard modification A can be made p-immune relative to B.

The construction can also be modified such that A is d-self-reducible relative to B: We choose for each length $n = tow(i)$ a random string x_i of length $4n$. Decompose $x_i = q_i r_i$ into two strings q_i, r_i of length $3n$ and n, respectively. Let $B = \{q_i, r_i : i \geqslant 0\}$ and $A = \{q_i u : i \geqslant 0,\ u \text{ is prefix of } r_i\}$. A is d-self-reducible relative to B (to simplify notation we have d-self-reducibility only w.r.t. some polynomially related well-ordering; of course we can also modify the definition of A such that it becomes d-self-reducible w.r.t. the classical length decreasing ordering). If $A \in \mathrm{P}^B$ then r_i is computable from q_i in polynomial time relative to B, contradicting the randomness of x_i. All strings not of the form $q_i u$ with $|u| \leqslant tow(i)$ have constant instance complexity w.r.t. A. Certainly, $ic^{t,B}(q_i u : A) \leqslant \log(|q_i|) + tow(i) + \mathrm{O}(1)$ for some polynomial t. On the other hand, $C^{t,B}(q_i u) \geqslant |q_i|/2 = (3/2)tow(i)$ for every polynomial t and all sufficiently large i. Thus, A does not have hard instances relative to B.

(c) Orponen et al. [18, Theorem 4.1] proved that for every self-reducible set $A \notin \mathrm{P}$ every polynomial t and every c, there are infinitely many x such that $ic^t(x : A) \geqslant c \log(|x|)$. To get a relativized world where this is tight we choose a suitable oracle of Homer and Longpré [8, Theorem 16]:

They show that for every $\overline{f}(n)$ such that $\overline{f}(n)/\log n$ is nondecreasing and unbounded there is an oracle B such that $\mathrm{P}^B \neq \mathrm{NP}^B$ and there is a set $C \in \mathrm{NP}^B$ which is $\leqslant^{p^B}_{\overline{f}(n)\text{-tt}}$-hard for NP^B. Furthermore, the queries of the tt-reduction only depend on the length of the input. Thus, for every $A \in \mathrm{NP}^B$ an A-consistent program which solves x only needs to know a program for the reduction, the length of x, and the $\overline{f}(|x|)$ many answers to the queries to C. This can be encoded in a string of length $2\log|x| + \overline{f}(|x|) + \mathrm{O}(1) = f(|x|) + \mathrm{O}(1)$ for a suitably chosen $\overline{f}$. □

We might still hope to prove Conjecture 11, the *CD*-version of Conjecture 9. However, also in this case we can construct a relativized counterexample. The construction may be of independent interest.

Theorem 21. *There is an infinite set B which contains only strings of length $tow(k)$ for all $k \geqslant 0$ and no other strings such that for every polynomial t:*

$$CD^{t,B}(x) \geqslant |x|/5 \quad \textit{for almost all } x \in B.$$

Proof. Let n_1 be a constant large enough such that the inequalities (1)–(3) below are satisfied. B is defined in stage k on $\Sigma^{tow(k)}$ as follows:

Let $B_k = \{x \in B : |x| < tow(k)\}$, $n = tow(k)$. If $n < n_1$ then let $B \cap \Sigma^{tow(k)} = \emptyset$. Otherwise, choose $2^{n/4}$ strings $x_1, \ldots, x_{2^{n/4}}$ of length n such that $C^{B_k}(x_i|x_j) \geqslant n/4$ for all $i \neq j$. For instance, let x be a random string of length $n2^{n/4}$ with $C^{B_k}(x) \geqslant |x|$, split it up into $2^{n/4}$ blocks of length n, and let x_i be the ith block. If $C^{B_k}(x_i|x_j) \leqslant n/4$ via p, $|p| \leqslant n/4$, then we could describe x by p, i, j and the shortened string x where the ith block is cut out. If we represent p, i, j in binary, each as a string of length $n/4$ (possibly with leading zeros), the concatenation σ of all these strings and of the shortened string x has length $3(n/4) + n(2^{n/4} - 1)$. Since n is uniquely determined by

$|\sigma|$, all four components can be recovered from σ. Thus we would get the contradiction

$$C^{B_k}(x) \leqslant 3(n/4) + n(2^{n/4} - 1) + O(1) < n2^{n/4} \quad \text{for all } n \geqslant n_1 \tag{1}$$

Assume that

$$2^n > n^{\log n} \quad \text{for all } n \geqslant n_1 \tag{2}$$

Let $I_n = \{x_1, \ldots, x_{2^{n/4}}\}$ and let P_n be the set of all $2^{n/5} - 1$ programs of length less than $n/5$. If we run any $p \in P_n$ on input $x_i \in I_n$ for at most $n^{\log n}$ steps with any oracle then, by choice of n_1, no string of length $\geqslant 2^n = tow(k+1)$ can be queried. If the oracle is $M = B_k \cup D$ with $D \subseteq I_n$ then no x_j with $j \neq i$ can be queried. Otherwise, consider the x_j which is queried first, say in step $s \leqslant n^{\log n}$. Then we can describe x_j from x_i by

$$|p| + 2\log s + O(1) \leqslant n/5 + 2(\log n)^2 + O(1) < n/4 \tag{3}$$

bits for $n \geqslant n_1$, which contradicts the hypothesis $C^{B_k}(x_i | x_j) \geqslant n/4$. For each $p \in P_n$ let T_p denote the set of all $x \in I_n$ such that $U^{B_k \cup \{x\}}(p,x) = 1$ in at most $n^{\log n}$ steps. By the remarks above,

$$\text{if } x \in D \subseteq I_n \text{ then the first } n^{\log n} \text{ steps in the computations of } U^{B_k \cup \{x\}}(p,x) \text{ and } U^{B_k \cup D}(p,x) \text{ are identical.} \tag{$*$}$$

Now we reduce the set I_n by the following procedure: Let $D = I_n, H = P_n$. If there is $x \in D$ and $p \in H$ such that $D \cap T_p = \{x\}$, then let $D = D - \{x\}$, $H = H - \{p\}$, and iterate the procedure.

Since each iteration decreases $|D|$ and $|H|$ by 1, and $|I_n| > |P_n|$, it follows that the procedure stops after finitely many steps with the final values D, H and $|D| > |H| \geqslant 0$. Define $B \cap \Sigma^n = D$.

For every $p \in P_n$, we argue that p is not a witness for the CD-complexity of any $x \in D$. If $p \in H$ then $|T_p \cap D| \neq 1$; so, using $(*)$, $U^B(p,-)$ does not accept a unique string from D. If $p \in P_n - H$, then no $x \in D$ belongs to T_p, i.e., $U^{B_k \cup \{x\}}(p,x)$ does not halt within $n^{\log n}$ steps with output 1. By $(*)$, $U^B(p,-)$, does not accept any string from D within $n^{\log n}$ steps.

Thus, the construction yields $CD^{t,B}(x) \geqslant |x|/5$ for all $x \in B$ and $t(n) = n^{\log n}$. Since B is infinite, the theorem follows. □

Corollary 22. *Conjecture 11 fails relative to some oracle.*

Proof. Choose B as in the previous theorem. There is a set $A \subseteq B$ such that A is recursive in B, $A \notin \mathrm{P}^B$, and for all k: Either $A \cap \Sigma^{tow(k)} = B \cap \Sigma^{tow(k)}$ or $A \cap \Sigma^{tow(k)} = \emptyset$. Let $t(n) = n^2$. Clearly, $ic^{t,B}(x : A) = O(1)$ for all $x \notin B$. If $x \in B, |x| = tow(k)$, consider the B-recursive program p which computes $\chi_B(z)$ for all $z \in \Sigma^{tow(k)}$ and otherwise outputs $\perp$. We can choose $|p| \leqslant k + O(1)$ and assume that the running time of p is bounded by t. Thus, $ic^{t,B}(x : A) \leqslant \log^*(|x|) + O(1)$ for all $x \in B$. Hence, by the choice of B, $ic^{t,B}(x : A) = o(CD^{t',B}(x))$ for every polynomial t'. □

4. The *ic*-measure is nonrecursive

Trivially, t-bounded Kolmogorov complexity is recursive for each recursive time bound t. However, since it is undecidable whether a t-bounded program is A-consistent, Orponen et al. [18, p. 103] conjectured that t-bounded instance complexity may be nonrecursive. This is confirmed by our next result.

Theorem 23. *Let t be a recursive time bound. There is a recursive set A such that $\lambda x.ic^t(x : A)$ is nonrecursive.*

Proof. Let $id = \lambda n.n + 1$. Let $\{p_i\}_{i \geqslant 0}$ be a recursive sequence of U-programs such that

$$U^{id}(p_i, x) = \begin{cases} 0 & \text{if } (\exists y)[x = \langle i, y\rangle]; \\ \perp & \text{otherwise.} \end{cases}$$

For simplicity we assume that the p_i are all $(n+1)$-time bounded; clearly there exist optimal interpreters with this property; in general one would have to replace $(n+1)$ by $c \cdot n \cdot \log n + c$ where the constant c depends on the interpreter.

Let $L_i = \{\langle i, x\rangle : x \in \Sigma^*\}$. Uniformly in i we define A on L_i as follows. Let i be fixed and let $x_0, x_1, \ldots$ be the listing of L_i according to the standard ordering. We will put at most one x_s into A. The goal is to diagonalize against the ith partial recursive function φ_i, i.e., we want to make sure that $\varphi_i(e) \neq ic^t(x_e : A)$ for some e. Let $\varphi_{i,s}(x)$ denote the result, if any, after s steps of computation of $\varphi_i(x)$.

Step 0 : Let $e_0 = 0$.
Step $m+1$: Search for the least $s > e_m$ such that $\varphi_{i,s}(e_m)\downarrow \leqslant |p_i|$.
Let $I = \{q : |q| \leqslant |p_i| \wedge (\forall k \leqslant m)[U^t(q, x_{e_k})\downarrow \neq \perp \Rightarrow U^t(q, x_{e_k}) = 0]\}$.
Let $I_m = \{q \in I : U^t(q, x_s) = 0\}$.
If there is $j < m$ such that $I_m = I_j$, then put x_s into A and halt.
Else let $e_{m+1} = s$ and goto step $m+2$.

A is recursive: Clearly A is r.e., but also $\overline{A}$ is r.e. since we have for $s > 0$:

$$x_s \notin A \cap L_i \Leftrightarrow (\exists m)[e_m \text{ is defined } \wedge e_m < s \leqslant e_{m+1} \wedge (\varphi_{i,s}(e_m)\uparrow \vee \varphi_{i,s}(e_m)\downarrow > |p_i|)].$$

Suppose for a contradiction that $\varphi_i(e) = ic^t(x_e : A)$ for all e.

If in some step $m+1$ the search does not terminate then $A \cap L_i = \emptyset$, thus $ic^t(x : A) \leqslant |p_i|$ for all $x \in L_i$ via program p_i. On the other hand, $\varphi_i(e_m)$ must be either undefined or greater than $|p_i|$, a contradiction. Thus in each step $m+1$ the search terminates.

Since there are only finitely many possible values for I_m the construction halts at some step $m_0 + 1$ where some x_s is put into A and $I_j = I_{m_0}$ for some $j < m_0$. Note that every A-consistent program q with $|q| \leqslant |p_i|$ and $U^t(q, x_{e_{j+1}}) = 0 = \chi_A(x_{e_{j+1}})$ is a member of I_j. But, by the action in step $m_0 + 1$, no member of $I_{m_0} = I_j$ is A-consistent. Thus $ic^t(x_{e_{j+1}} : A) > |p_i| \geqslant \varphi_i(e_{j+1})$, a contradiction.

Hence it follows that $\lambda x.ic^t(x : A)$ is not a recursive function. □

5. *C* versus *CD*

In the previous sections we have compared instance complexity and Kolmogorov complexity. Since *CD*-complexity can be seen as a special case of instance complexity, it is natural to investigate the connection between *C*- and *CD*-complexity. We consider the question whether, with respect to polynomial time bounds, the *C*-complexity can be bounded by the *CD*-complexity. This is formally stated by the following hypotheses.

(H1) For every polynomial t there is a polynomial t' and a constant c such that for all $x, y : C^{t'}(x|y) \leqslant CD^{t}(x|y) + c$.

(H2) For every polynomial t there is a polynomial t' and a constant c such that for all $x : C^{t'}(x) \leqslant CD^{t}(x) + c$.

The promise problem (1SAT,SAT) belongs to P if (by definition) there is a deterministic polynomial-time algorithm which accepts all Boolean formulas with a unique satisfying assignment, and rejects all Boolean formulas which are not satisfiable. (1SAT, SAT) is complete for $\mathcal{UP}$, the promise version of UP (see [5] for more information on this topic).

Theorem 24. (H1) $\Leftrightarrow$ (1SAT,SAT) $\in$ P.

Proof. ($\Leftarrow$) Let t be a fixed polynomial. If there is a polynomial time algorithm for (1SAT,SAT) then we can determine in polynomial time for each t-time bounded program p, each y, and each n, the unique $x \in \Sigma^n$ such that $U(p, y, x) = 1$, if such an x exists. Now (H1) follows easily.

($\Rightarrow$) We assume that assignments a of a Boolean formula ϕ are padded such that $|a| = |\phi|$. There is a program p and a polynomial t such that for every Boolean formula ϕ and assignment a:

$$U(p, \phi, a) = \begin{cases} 1 & \text{if } \phi(a) = \mathit{true}; \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for every Boolean formula with exactly one satisfying assignment a^* we get $CD^{t}(a^*|\phi) \leqslant |p|$. By hypothesis there is a polynomial t' and a constant c (independent of ϕ) such that $C^{t'}(a|\phi) \leqslant c$. Thus, we get a polynomial time algorithm for (1SAT,SAT): On input ϕ, we simulate $U(p', \varepsilon, \phi)$ for at most $t'(|\phi|)$ steps for all p' with $|p'| \leqslant c$. We accept only if one of them outputs a satisfying assignment of ϕ. Since only a constant number of programs is simulated, the computation runs in polynomial time. □

Trivially, (H1) implies (H2), but the converse might fail. However, we have the following partial converse.

Proposition 25. (H2) $\Rightarrow$ FewP $\cap$ SPARSE $\subseteq$ P.

Proof. Let A be a sparse set and let M be a nondeterministic Turing machine that accepts A. Let $|A \cap \Sigma^n|$, the running time of M, and the number of accepting paths all

be bounded by a polynomial. For each n let

$$l_n = \langle x_1, w_{1,1}, \ldots, w_{1,m_1}, \ldots, x_s, w_{s,1}, \ldots, w_{s,m_s} \rangle$$

be the list of all elements $x_1 < \cdots < x_s$ in $A \cap \Sigma^n$, where $w_{i,1}, \ldots, w_{i,m_i}$ is the list of all accepting paths for x_i in lexicographical ordering (we assume that $|w_{i,j}| > n$). Note that l_n can be uniquely recognized in polynomial time if we are given n and $m_1 + \cdots + m_s$. Thus, $CD^t(l_n) \leqslant l(\langle n, m_1 + \cdots + m_s \rangle) = O(\log n)$ for some polynomial t. Using (H2) it follows that $C^{t'}(l_n) = O(\log n)$ for all n and some polynomial t'. This means that we can generate all elements in $A \cap \Sigma^n$ in polynomial time, hence $A \in P$. In fact, A is even p-printable (cf. [13, Definition 7.13]). □

6. Open questions

In this paper we have answered many of the open questions from the literature concerning instance complexity. However, new questions turned up which we recommend for further study:

(1) Is there a complexity theoretic characterization of when every recursive set $A \notin P$ has p-hard instances? Does GI, the graph isomorphism problem, have p-hard instances unless GI $\in$ P?

(2) Resource bounded Kolmogorov complexity turned out to have many applications in structural complexity theory (see [13, Ch. 7]). We believe that our instance complexity results and the techniques used in this paper should also have important consequences in complexity theory.

(3) Does (H2) imply (H1) as defined in Section 5, or is there a relativized counterexample?

Acknowledgements

We would like to thank Elvira Mayordomo for the remark following Corollary 15, Marcus Schäfer for proofreading, and the two referees for corrections and detailed suggestions on how to improve the presentation.

References

[1] A. Amir and W. I. Gasarch, Polynomial terse sets, *Inform. Comput.* **70** (1988) 37–56.

[2] J. Balcázar, J. Díaz and J. Gabarró, *Structural Complexity I* (Springer, Berlin, 1988).

[3] J. Balcázar, J. Díaz and J. Gabarró, *Structural Complexity II* (Springer, Berlin, 1990).

[4] H. Buhrman and P. Orponen, Random strings make hard instances, in: *Proc. Structure in Complexity Theory, 9th Annual Conf.*, (1994) 217–222.

[5] J. Cai, L.A. Hemachandra and J. Vyskoč, Promises and fault-tolerant database access, in: *K. Ambos-Spies, S. Homer and U. Schöning,* eds., *Complexity Theory* (Cambridge University Press, Cambridge, 1993) 227–244.

[6] J. Hartmanis, Generalized Kolmogorov complexity and the structure of feasible computation, in: *Proc. 24th IEEE Symp. Foundations of Computer Science* (1983) 439–445.
[7] J. Hartmanis, On sparse sets in NP−P, *Inform. Process. Lett.* **16** (1983) 55–60.
[8] S. Homer and L. Longpré, On reductions of NP sets to sparse sets, *J. Comput. System Sci.* **48** (1994) 324–336.
[9] K. Ko, On the notion of infinite pseudorandom sequences, *Theoret. Comput. Sci.* **48** (1986) 9–33.
[10] K. Ko, A note on the instance complexity of pseudorandom sets, in: *Proc. Structure in Complexity Theory, 7th Annual Conf.*, (1992) 327–337.
[11] K. Ko, P. Orponen, U. Schöning and O. Watanabe, What is a hard instance of a computational problem? in: A. Selman, ed., *Structure in Complexity Theory*, Lecture Notes in Computer Science 223 (Springer, Berlin, 1986) 197–217.
[12] M. Kummer, Kolmogorov complexity and instance complexity of recursively enumerable sets, *SIAM J. Comput.*, to appear.
[13] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications* (Springer, New York, 1993).
[14] L. Longpré and O. Watanabe, On symmetry of information and polynomial time invertibility, Manuscript, 1993.
[15] E. Mayordomo, Almost every set in exponential time is p-bi-immune, *Theoret. Comput. Sci.* **136** (1994) 487–506.
[16] A. Naik, M. Ogiwara and A. Selman, P-selective sets, and reducing search to decision vs. self-reducibility, in: *Proc. Structure in Complexity Theory, 8th Annual Conf.*, (1993) 52–64.
[17] P. Orponen, On the instance complexity of NP-hard problems, in: *Proc. Structure in Complexity Theory, 5th Annual Conf.*, (1990) 20–27.
[18] P. Orponen, K. Ko, U. Schöning and O. Watanabe, Instance complexity, *J. Assoc. Comput. Mach.* **41** (1994) 96–121.
[19] C.H. Papadimitriou, *Computational Complexity* (Addison-Wesley, Reading, MA, 1994).
[20] A. Selman, P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP, *Math. Systems Theory* **13** (1979) 55–65.
[21] M. Sipser, A complexity theoretic approach to randomness, in: *Proc. 15th ACM Symp. Theory of Computing* (1983) 330–335.
[22] O. Watanabe, On one-one polynomial time equivalence relations, *Theoret. Comput. Sci.* **38** (1985) 157–165.